A Simple Survey on Diffusion Models for Image Synthesis Since 2020

Zhixin Ling

ABSTRACT

Image Synthesis is a basic research task in computer vision. In the early deep learning era, GANs, VAEs, ARs and flow models are comprehensively adopted for this task. Recently, diffusion models (DMs) are shown to be able to generate various high-quality images. Hence, the relevant research on DM has become a hotspot. In this paper, we present a simple survey on DMs since 2020. We focus on the application of Image Synthesis and closely related works. We will show HOW these diffusion models work but WHY is not our focus. Detailed mathematical derivations will be skipped. Similarities and differences of these works will be emphasized.

ACM Reference Format:
Zhixin Ling. 2022. A Simple Survey on Diffusion Models for Image Synthesis Since 2020. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Our paper mainly consists of three parts:

(1) Horizontal Comparison (Sec.2). We briefly look through deep image synthesis methods and show their advantages and differences.
(2) Vertical Comparison (Sec.3). We present comparisons between diffusion models since 2020. We are going to discuss the differences between their motivations in a high-level view. Also, their open-source repositories and datasets will be summarized.
(3) Conclusion (Sec.4). We conclude our investigation on diffusion models and present some ideas for future research.

2 HORIZONTAL COMPARISON

Neural networks are able to learn various functions, but do NOT directly learn probabilistic distributions. As a result, deep generative models parameterize various complex probabilistic distributions and learn the distribution parameters, *e.g.*, mean and variance of a Gaussian distribution.

Deep image synthesis models can be roughly categorized into Five groups: generative adversarial networks (GANs), autoregressive models (ARs), flows, variational autoencoders (VAEs), diffusion models (DMs). Their main differences are listed as follows:

(1) VAE [7] encodes an input image into latent variables, which follows a Gaussian distribution.
(2) GAN [3] adopts a generator to synthetize an image and a discriminator to distinguish a true image from a false one.
(3) Flow [2] uses a reversible function f to encode an input image into latent variables and decode the latent variables via reversed f.
(4) AR [15] generates an image pixel by pixel, like tangling a next-word prediction task in NLP.
(5) DM [5] learns decodes latent variables into an image step by step. In each step, new latent variables are sampled from a Gaussian distribution that is depended on the current latent variables.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

3 VERTICAL COMPARISON

In these part, we introduce different diffusion models. We try to skip heavy mathematical theorems and directly go to research conclusions. Also, for simplicity, we will ignore some unimportant details, *e.g.*, a constant scale parameter.

3.1 DDPM

Denoising Diffusion Probabilistic Model(DDPM)[5] is a pioneer work. To make this work clear is crucial.

3.1.1 Forward Process. DDPM encodes an given image x_0 into Gaussian noises x_T step by step, which is named a **forward** process. According to the chain rule, we explain the process in formula:

$$\begin{aligned} q(x_t|x_{t-1}) =& \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}); \\ q(x_T|x_0) =& \prod_{t=1}^{T} q(x_t|x_{t-1}). \end{aligned} \tag{1}$$

where β_t ranges from 0 to 1 and grows with t. Therefore, Gaussian noises employed on x_t step by step. At last, $q(x_T|x_{T-1})$ obeys a Gaussian distribution that is not related with x_{T-1} at all.

3.1.2 Backward Process. The decoding stage, also a **backward** process, is performed inversely:

$$\begin{aligned} p(x_T) =& \mathcal{N}(0, \mathbf{I}); \\ q(x_{t-1}|x_t) =& \mathcal{N}(\mu(x_t, t), \Sigma(x_t, t)); \\ p(x_0) =& p(x_T) \prod_{t=T}^{0} q(x_{t-1}|x_t). \end{aligned} \tag{2}$$

In this process, μ and Σ are modeled by neural networks. The backward process starts from $\mathcal{N}(0, \mathbf{I})$. In each step, a Gaussian distribution is deriviated according to the current noised image x_t. The the $(t-1)$-step image is sampled from the new distribution. Therefore, the image synthesis process is actually converted into a denoising process, just as indicated by the model title **Denoising** Diffusion Probabilistic Model.

In test stage (*i.e.*, synthesis stage), the backward process is used for image synthesis.

Basic Model	A Simplified Naive Loss	Generation Example
VAE [7] (ICLR'14)	$\mathcal{L}_{KL} = KL(q(\mathbf{z}\|x)\|\|p(\mathbf{z})), \mathcal{L}_{rec} = \|\|p(x\|\mathbf{z}) - p(x)\|\|^2, p(\mathbf{z}) = \mathcal{N}(0,1)$	$\hat{x} = p(x\|\mathbf{z}), \mathbf{z} \sim \mathcal{N}(0,1)$
GAN [3] (NIPS'14)	$\mathcal{L}_D = \log D(x) + \log(1 - D(G(\mathbf{z}))), \mathcal{L}_G = \log(D(G(\mathbf{z}))), \mathbf{z} \sim \mathcal{N}(0,1)$	$\hat{x} = G(\mathbf{z}), \mathbf{z} \sim \mathcal{N}(0,1)$
Flow [2] (ICLR'14)	$\mathcal{L}_{flow} = \|\|x - f(\mathbf{z})\|\|^2 + \|\|\mathbf{z} - f^{-1}(x)\|\|^2, \mathbf{z} \sim \mathcal{N}(0,1)$	$\hat{x} = f(\mathbf{z}), \mathbf{z} \sim \mathcal{N}(0,1)$
AR [15] (PMLR'16)	$\mathcal{L} = \|\|p(x\|\mathbf{z}) - p(x)\|\|^2, p(x) = \prod_{i=1}^{w\times h} p(x_i\|x_1, x_2, \cdots, x_{i-1}), p(x_1) = \mathcal{N}(0,1)$	$x_i = \prod_{i=1}^{w\times h} p(x_i\|x_1, x_2, \cdots, x_{i-1}), x_1 \sim \mathcal{N}(0,1)$
DM [5] (NIPS'20)	$\mathcal{L}_t = \|\|f(\sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon') - \epsilon'\|\|^2, \epsilon \sim \mathcal{N}(0,1)$	$x_t \sim \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}), \alpha_t = \prod_{s=1}^{t} 1 - \beta_s$

Table 1: This table follows the naming convention in the original papers. p/q originally stands for a distribution. For simplicity, we also interpret p/q as a function in this table. For example, $\mathcal{L}_{rec} = ||p(x|\mathbf{z}) - p(x)||^2$ is equivalent to $\mathcal{L}_{rec} = \mathbb{E}_{x\sim p(x), \mathbf{z}\sim q(\mathbf{z}|x), x'\sim p(x'|\mathbf{z})}||x' - x||^2$. Both of p and q along with G, D, f can be modeled by a neural network.

3.1.3 The Close Form of Forward Process. In the backward process, μ and Σ should be inferred via neural networks. However, in the forward process, careful readers should have noticed that, $q(x_t|x_0)$ is actually a combination of a sequence of known Gaussian distributions $\mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I})$. Therefore, $q(x_t|x_0)$ also obeys a known Gaussian distribution:

$$\begin{aligned} \alpha_t =& 1 - \beta_t; \\ \overline{\alpha_t} =& \prod_{s=1}^{t} \alpha_s; \\ q(x_t|x_0) =& \mathcal{N}(\sqrt{\overline{\alpha_t}}x_0, (1-\overline{\alpha_t})\mathbf{I}). \end{aligned} \tag{3}$$

3.1.4 Training. The training loss is simple:

$$\mathcal{L} = ||E(\sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon, t) - \epsilon||^2. \tag{4}$$

E is a noise estimator, usually a U-Net conditioned by t. E aims to predict Gaussian noises ϵ.

3.1.5 Some Details. Σ is simply assumed as scaled $\mathbf{I}$ in pratical, while μ is modeled as follow:

$$\mu(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}E(x_t, t)). \tag{5}$$

In the training stage, a step $t \in [1, T]$ is randomly sampled to otain the loss $\mathcal{L}$.

The generation process must be conducted step by step. The sugguested diffusion steps T should be at least 50 to obtain an okay 128×128 image. For the best image quality, T is usually set to 1000. It indicates that E should be run 1000 times for a generation, consuming about 100 seconds using V100.

3.2 GD

Guided Diffusion (GD) [1] mainly introduces three improvements: **learnable** Σ, **synthesis condition** and **architectural improvement**.

3.2.1 learnable Σ. Instead of fixing Σ at $\mathbf{I}$, GD defines a learnable Σ:

$$\begin{aligned} \tilde{\beta}_t =& \frac{1-\alpha_{t-1}}{1-\alpha_t}\beta_t, \\ \Sigma(x_t, t) =& \exp(v \log \beta_t + (1-v)\log \tilde{\beta}_t), \end{aligned} \tag{6}$$

where v is a learnable parameter. This improvement refines the image and reduces inference time.

3.2.2 Learnable Σ. To obtain images that meet the need of users, we want to add a condition in the synthesis process. For example, the condition is image category constraint. Instead of naively adding a condition input to the U-Net, GD appends a condition head (*i.e.*, a classification head) to the U-Net output. GD train the classification head by classifying the synthetized images, and does NOT need to fine-tune the U-Net parameters. In the test stage, the image synthesis procedure is formulated below:

$$\begin{aligned} g =& \Delta_{x_t} \log(p(y|x_t)), \\ x_t \sim& \mathcal{N}(\mu + \Sigma g, \Sigma), \end{aligned} \tag{7}$$

where $p(y|x_t)$ is a classification CNN. Besides, it should be noticed that U-Net consumes all trainable parameters in DDPM. Therefore, for different conditions, only different condition heads are required.

We use an example to intuitively explain Eq.7. Let our target be a panda image. The original DDPM is able to provide a panda image. However, the chance is poor because of the randomness in Gaussian sampling. Therefore, we "move" the μ to a certain direction that results in a higher propability to provide a panda image. And the gradient g tells what the direction is.

3.2.3 Architectural improvement. The architectural improvements includes:

(1) Increase depth, decreases width.
(2) Increase attention heads.
(3) Single-resolution attention → multi-resolution attention.
(4) For upsampling and downsampling, interpolation → residual blocks.
(5) Rescaling residual connections with $\frac{1}{\sqrt{2}}$ (not clear yet).

3.3 SGD

Semantic Guidence Diffusion [9] (SGD) is an extension of GD. SGD studies several specific guidances: **language guidance**, **image guidance** and **multimodal guidance**. SGD first generalize the GD condition function of $p(y|x_t)$ to $F(x_t, y, t)$. For different guidances, F is instantiated to a special form.

3.3.1 Language guidance: $F(x_t, l, t) = F''(x_t, t) \cdot F'(l)$. E' encode x_t into an image feature, while $F'(l)$ encode text l into a text feature. The similarity between $F''(x_t, t)$ and $F'(l)$ is measured via dot product.

3.3.2 Image guidance: there are three different image guidances.

The first is semantic-level guidance:

$$F(x_t, x_t', t) = F'(x_t, t) \cdot F'(x_t', t), \tag{8}$$

workflow step	training workflow	test workflow
0	input: caption y, image x	input: caption y
1	CLIP produces CLIP text embedding z_t, CLIP image embedding z_i	CLIP produces CLIP text embedding z_t
2	train a diffusion prior model (DPM) that generates a prior $P(z_i\|y)$ and minimize $\mathbb{E}_{z_i'\sim P(z_i'\|y)}\|\|z_i' - z_i\|\|^2$	DPM generates z_i
3	train a diffusion decoder model (DDM) that inverts z_i into x and minimize $\mathbb{E}_{x'\sim P(x'\|z_i,y)}\|\|x' - x\|\|^2$	DDM inverts z_i into x

Table 2: A simplified DALLE2 training and test workflow.

where x_t' is the reference image noised by Eq. 3. It works similarly with language guidance.

The second is structure-level guidance:

$$F(x_t, x_t', t) = -\sum_j \frac{1}{C_j H_j W_j} ||F'(x_t, t)_j - F'(x_t', t)_j||, \quad (9)$$

where $F'()_j \in \mathcal{R}^{C_j \times H_j \times W_j}$.

The third is style-level guidance:

$$F(x_t, x_t', t) = -\sum_j ||Gram(x_t, t) - Gram(x_t', t)||_F^2, \quad (10)$$

where $Gram$ yields a Gram matrix and $||\cdot||_F$ is Frobenius norm. For the Gram matrix, please refer to https://blog.csdn.net/wangyang20170901/article/details/79037867. For the Frobenius norm, please refer to https://blog.csdn.net/qq_27946691/article/details/56488063.

3.3.3 Multimodel guidance: $F = \sum_{i=1}^{g} w_i F_i$, where F_i is the $i-th$ model guidance and w_i is a weight. In other words, the multimodel guidance is simply a sum of guidances from different models.

3.4 CFDG

Classifier-Free Diffusion Guidence [6] (CFDG) argues that **1)** GD/SGD is NOT able to jointly train the U-Net and the condition head; **2)** the condition head largely increases inference computation overhead because of the requirement of gradient Δ. Therefore, CFDG integrates the condition into the noise estimator E in Sec. 3.1 and obtains E':

$$E'(x_t, t, y) = E(x_t, t, null) + scale(E(x_t, t, y) - E(x_t, t, null)). \quad (11)$$

However, the whole noise estimator must be re-trained for different condition types.

3.5 GLIDE

GLIDE [10] replaces the category condition of CFDG with language condition (*i.e.*, image caption):

$$E'(x_t, l, l) = E(x_t, t, null) + scale(E(x_t, t, l) - E(x_t, t, null)). \quad (12)$$

Language embedding l is derived from a CLIP model[11].

3.6 DALLE

Although DALLE[13] does not adopt any diffusion model, it is the beginning DALLE generation. The paper is filled with engineering details and is good for future reference. Briefly, there are two stages in DALLE:

(1) DALLE encodes the image into image tokens via a discrete VAE (dVAE);
(2) An AR model is adopted to learn the joint distribution of image and text tokens.

More specifically, DALLE tries to maximize a evidence lower bound:

$$ELB = \mathbb{E}_{z\sim q_\phi(z|x)}(\ln p_\theta(x|y,z) - KL(q_\phi(y,z|x), p_\varphi(y,z))). \quad (13)$$

Here, ϕ is the dVAE encoder; θ is the dVAE decoder; φ is the AR model. x is an images. y is the caption y. z is the image token.

3.7 DALLE2

3.7.1 Two main modules of DALLE2: CLIP image prior and DM.. DALLE2[12] incorperates strengths of CLIP and GLIDE, also consisting of two main modules:

(1) A CLIP image prior $P(z_i|y)$ that produces CLIP[11] image embeddings z_i conditioned on captions y;
(2) A diffusion decoder $P(x|z_i, y)$ that produces images x conditioned on CLIP image embeddings z_i (and optionally captions y).

The DALLE2 workflows is shown in Tab. 2. Note that we will skip CLIP details [11] and only discuss the diffusion decoder..

3.7.2 The diffusion prior model (DPM).. DPM learns $P(z_i|y)$. Ramesh *et. al.* also tries AR to learn $P(z_i|y)$ but find out that the diffusion model is better. The noise estimator of the DPM is a transformer decoder with a causal attention mask. The input sequence consists of: **1)** encoded text l, **2)** the CLIP text embedding z_t, **3)** the diffusion timestep τ and **4)** the noised CLIP image embedding z_i^τ. Different from common DMs, the output directly predicts the *unnoised CLIP image embedding*, instead of the noise. We explain the DPM loss in formula:

$$\mathcal{L}_{prior} = \mathbb{E}_{\tau\sim[1,T], z_i^\tau\sim q_\tau}||f(z_i^\tau, \tau, l, z_t) - z_i||^2. \quad (14)$$

f is the transformer decoder, q_t is a Gaussian distribution of time step τ. The specific form of q_t is not given in the original paper, and we conjure that it should be similar to Eq. 3.

3.7.3 The diffusion decoder model (DDM).. DDM learns $P(x|z_i, y)$ and is similar to GLIDE. The DALLE2 architecture differs from GLIDE (Eq. 12) mainly in two apects:

(1) The CLIP image embedding z_i is projected and added to a timestep embedding.
(2) The CLIP image embedding z_i is projected into four extra tokens. These tokens are concatenated to output sequence from GLIDE text encoder.

4 CONCLUSION

4.1 Our contributions.

In this paper, we have discussed key ideas of several advanced diffusion models[1, 5, 6, 9, 12] and several other related generative works[2, 3, 7, 10, 13, 15]. Diffusion models take the task of image

Model	test cost	training cost for N condition types	text2image	dataset
DDPM [5]	noise estimation	noise estimation (condition is not allowed)	×	CIFAR10, LSUN
GD [1]	noise estimation + condition gradient derivation	noise estimation + condition derivation $\times N$	×	ImageNet
SGD [9]	noise estimation + condition gradient derivation	noise estimation + condition derivation $\times N$	√	FFHQ, LSUN
CFDG [6]	noise estimation $\times 2$	noise estimation $\times N$	√	ImageNet
GLIDE [6]	noise estimation $\times 2$ + CLIP inference	noise estimation $\times N$	√	a big mixture
DALLE2 [12]	noise estimation $\times(2+2)$ + CLIP inference	noise estimation $\times N$	√	a big mixture

Table 3: Resource comparisons. By marking "text2image" as ×, we indicate that the model is not originally proposed for the text-to-image task, but they might be transferred to this task after a proper modification.

Model	repository
DDPM [5] (NIPS'20)	Tensorflow Code https://github.com/hojonathanho/diffusion PyTorch Unofficial Code https://github.com/pesser/pytorch_diffusion
FastDPM [8] (arxiv'21)	Demo https://fastdpm.github.io. PyTorch Code https://github.com/FengNiMa/FastDPM_pytorch
VQ-Diffusion [4] (arxiv'21)	PyTorch Code https://github.com/microsoft/VQ-Diffusion
CFDG [6] (NIPS'21)	Unavailable
CLIP [11] (PMLR'21)	PyTorch Code https://github.com/OpenAI/CLIP
DALLE [13] (PMLR'21)	PyTorch Code for discrete VAE model only. https://github.com/openai/DALLE
v-diffusion sampling	PyTorch Code https://github.com/crowsonkb/v-diffusion-pytorch
DALLE2 [12] (arxiv'22)	A waitlist to reach the demo. https://labs.openai.com/waitlist
DDIM [14] (ICLR'21)	PyTorch Code https://github.com/ermongroup/ddim
GD [1] (NIPS'21)	PyTorch Code https://github.com/openai/guided-diffusion
GLIDE [10] (arxiv'21)	PyTorch Code https://github.com/openai/glide-text2im
SGD [9] (arxiv'21)	Project Page https://xh-liu.github.io/sdg/. Code is unavailable yet.

Table 4: Code repository overview. The code supports training and testing by default. Besides, all of the aboved papers can be reached in Arxiv.

synthesis as a denoising process. In text2image diffusion works[6, 9, 12], the image caption is taken as various conditions to guide the denoising process. Among these works, DALLE2[12] is able to incorperates the strengths of both CLIP and GLIDE to generate high-quality images. We also compare required resources in Tab. 3 and summarize code repositories in Tab. 4 for further study of readers.

4.2 Weaknesses.

Due to time limitation, we failed to include several other diffusion models into our paper[4, 8, 14]. This will be done in the future.

4.3 Future research directions.

Text2image works is now able to produce high-quality images, in which **1)** the objects have clear outlines and reasonable appearances; **2)** object relations can match the text description. Therefore, their might be an approach to produce a object mask that can tell different objects, or tell the foreground object so that the model can be adopted to more comprehensively researches or applications.

Existing works are not sensitive to numbers. For example, a text of "two cars are running on the street" might not be able to result in an image of two cars. On the contrary, humans can well distinguish two cars from one or three. To enhance numerical sensitivity might be a good research direction.

Step-by-step sampling is time-consuming. Efficiency is thus not an advantage of diffusion models. To reduce inference time will also be a promising research direction. It could be accomplished by reducing sampling steps or a coarse-to-fine generation framework,

REFERENCES

[1] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems* 34 (2021).

[2] Laurent Dinh, David Krueger, and Yoshua Bengio. 2014. Nice: Non-linear independent components estimation. *ICLR workshop* (2014).

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).

[4] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. 2021. Vector quantized diffusion model for text-to-image synthesis. *arXiv preprint arXiv:2111.14822* (2021).

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems* 33 (2020), 6840–6851.

[6] Jonathan Ho and Tim Salimans. 2021. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications.*

[7] Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. *ICLR* (2014).

[8] Zhifeng Kong and Wei Ping. 2021. On fast sampling of diffusion probabilistic models. *arXiv preprint arXiv:2106.00132* (2021).

[9] Xihui Liu, Dong Huk Park, Samaneh Azadi, Gong Zhang, Arman Chopikyan, Yuxiao Hu, Humphrey Shi, Anna Rohrbach, and Trevor Darrell. 2021. More

Control for Free! Image Synthesis with Semantic Diffusion Guidance. *arXiv preprint arXiv:2112.05744* (2021).

[10] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. 2021. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741* (2021).

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PMLR, 8748–8763.

[12] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. *arXiv preprint arXiv:2204.06125* (2022).

[13] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *International Conference on Machine Learning*. PMLR, 8821–8831.

[14] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising diffusion implicit models. (2021).

[15] Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. 2016. Pixel recurrent neural networks. In *International conference on machine learning*. PMLR, 1747–1756.

现有Diffusion总结

我们总结了14个近期用diffusion模型做图像生成的工作，对于开源工作，我们都测试了模型的生成效果。

1. 先说结论

14个近期的diffusion模型中，有5个模型是text2image的，开源的有4个。在这四个模型中，LDM生成的卡通漫画风图片可能是唯一一个合适做广告素材的，生成图片有美感，而且主体突出。

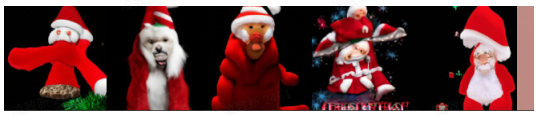
对于写实风格的素材，现有模型都不合适，要么主体不完整，要么图片无美感。原因由大到小可能包括：

a. 生成模型本身问题，模型本身并不针对广告素材
b. 文本先验提取得不好（比如CLIP先验对于广告素材描述不敏感）
c. 数据集问题。比如COCO中详细一点的图片描述都是结合场景的，而不是针对独立的物体。

一个可行的解决方案是用文本-素材数据集进行微调。数据集也可以考虑利用Image Caption模型自动生成一部分。

2. 图像生成的Diffusion工作一览表

模型名称	论文、代码等链接	论文主要内容 + (效果点评)	生成效果
text2image diffusion模型(开源)			

GLIDE [11] (arxiv’ 21)	Paper: https://arxiv.org/pdf/2112.10741.pdf PyTorch Code: https://github.com/openai/glide-text2im	GLIDE类似与GD，模型规模更大，而且在图像编辑任务上进行了探索。 对于真实物体，效果稍好，大概有10%可以得到完整的可以做素材图片。动漫人物生成效果很差。	 A picture of a car without backgrc  A picture of Santa Claus without back
LDM [16] (arxiv’ 21)	Paper: https://arxiv.org/pdf/2112.10752.pdf Pytorch code: https://github.com/CompVis/latent-diffusion	该模型也致力于降低扩散模型的时间成本。把图片编码到一个隐空间中，然后对该隐变量用扩散模型去处理。 对于真实物体，生成物体不完整，或者有背景。动漫人物生成效果较好。	 A picture of a car without backgrc  A picture of Santa Claus without back
VQ-Diffusion [4] (arxiv’ 21)	Paper: https://arxiv.org/pdf/2111.14822.pdf PyTorch Code: https://github.com/microsoft/VQ-Diffusion Paper sharing: https://www.bilibili.com/video/BV13Y4y1r7CH?spm_id_from=333.1007.top_right_bar_window_dynamic.content.click	先对图像用vq-vae压缩（这个步骤和dalle类似），因为图像已经离散化了，所以用mask和replace操作添加噪声。 写实物体的主体很突出，但是不美观而且不完整。可以生成少量美观的动漫风格的图片，但是多数也不完整，可能不满足广告素材需求。	 A picture of a car without backgrc  A picture of Santa Claus without back
DiscoDiffusion	Paper: based on CLIP[13] and GD[1]. PyTorch test Code https://github.com/alembics/disco-diffusion Demo: https://colab.research.google.com/github/alembics/disco-diffusion/blob/main/Disco_Diffusion.ipynb	谷歌的text2image模型，GD这篇文章很类似，有详细的代码。 生成一些浪漫主义、赛博朋克的场景，该模型很拿手，但是对于稍微写实的画风，该模型无能为力。也可参考知乎讨论 https://www.zhihu.com/question/530767425/answer/2520938410。	 A beautiful painting of a singular lighthouse, ..., yellow color scheme"  A picture of a car without background, yellow color scheme  A painting of Santa Claus

text2image diffusion模型(未开源)			
DALLE2 [14] (arxiv’22)	Paper: https://arxiv.org/pdf/2204.06125.pdf A waitlist to reach the demo: https://labs.openai.com/waitlist	用一个prior模块使文本编码转换为图像编码，再用一个decoder生成最终的图像。这个prior和decoder都是diffusion模型。三个子模块都可以分别训练。 dalle2的不足：易将物体和属性混淆，对于将文本放入图像中的能力不足。	
其他条件图像生成的diffusion工作(开源)			
DDPM [5] (NIPS’20)	Paper: https://proceedings.neurips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf Tensorflow Code: https://github.com/hojonathanho/diffusion PyTorch Code: https://github.com/pesser/pytorch_diffusion	扩散模型在图像生成任务上的开山之作。扩散模型通常包括两个过程，从信号逐步到噪声的正向过程/扩散过程，和从噪声逐步到信号的逆向过程。这两年后续的扩散模型大都是基于DDPM的框架所设计。 church生成效果较佳，不过第三幅图的文字有点奇怪。bedroom色调暗黄（对比DDIM）。	 LSUN-church  LSUN-bedroom
FastDPM [9] (arxiv’21)	Paper: https://arxiv.org/pdf/2106.00132.pdf Demo: https://fastdpm.github.io. PyTorch Code: https://github.com/FengNiMa/FastDPM_pytorch	主要目标：减少扩散步数，优化推理时间。主要贡献：1. 把离散的扩散步数泛化为连续的步数。2. 设计了连续扩散步数和连续噪声水平的一一映射。 人物依然是个问题，church第三幅图的人物没法看。生成bedroom的环境暗黄（对比DDIM）。	 LSUN-church  LSUN-bedroom
DDIM [18] (ICLR’21)	Paper: https://arxiv.org/pdf/2010.02502.pdf PyTorch Code: https://github.com/ermongroup/ddim	基于DDPM，提出一个非马尔可夫过程进行推测，能提高十倍的推测速度。不过训练同DDPM。 church第二幅图很奇怪，多数效果尚可。bedroom的生成效果较佳。	 LSUN-church  LSUN-bedroom

GD [1] (NIPS’21)	Paper: https://proceedings.neurips.cc/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf PyTorch Code https://github.com/openai/guided-diffusion	提出加入类别条件的一种方法：在噪声估计网络后面添加分类器，测试的时候用分类器梯度去引导生成生成图像的类别。 生成动物尚可，但是脸不行。我们对脸的质量要求较高，这里生成的人脸都很别扭。	 狒狒 菜锅 蛇 长体蛇鲔 和服
IDDPM [12] (ICML’21)	Paper: http://proceedings.mlr.press/v139/nichol21a/nichol21a.pdf PyTorch code: https://github.com/openai/improved-diffusion	DDPM的IS和FID都很好，不过likelihood指标较低。IDDPM做了很多实验对这个问题进行了探究，并提出了改进。此外，IDDPM也探究了通过减少采样来提高模型效率的方式。 这里展示的bedroom是64x64分辨率的，效果还行。ImageNet是256x256的，有很多畸形的物体。	 LSUN-bedroom  ImageNet
其他条件图像生成的diffusion工作(未开源)			
CFDG [7] (NIPS’21)	Paper: https://openreview.net/pdf?id=qw8AKxfYbI	提出了一个Classifier-Free Diffusion Guidance的方法，是主流的添加diffusion条件的方式。本质上是训练一个同时支持有条件和无条件噪声估计的模型。在逆行过程中，结合有条件和无条件噪声估计得到结果。	
SGD [10] (arxiv’21)	Paper: https://arxiv.org/pdf/2112.05744.pdf Project Page https://xh-liu.github.io/sdg/.	GD只是提出了类别引导的方式，SGD则细化了文本引导、图片引导和风格引导等引导方式。	
CDM [6] (JMLR’21)	Paper: https://cascaded-diffusion.github.io/assets/cascaded_diffusion.pdf Project page: https://cascaded-diffusion.github.io	CDM级联了多个扩散模型。其中，第一个扩散模型是常规的生成图像的扩散模型，后续的是超分辨率扩散模型。	
其他image diffusion相关工作			

Palette [16] (arxiv’ 21)	Paper: https://arxiv.org/pdf/2111.05826.pdf PyTorch code: https://github.com/Janspiry/Palette-Image-to-Image-Diffusion-Models	这是一个图像到图像的扩散模型，也做了inpainting的任务。此外，文章还提出了一套新的图像质量评估方法。	